

frivol

Generated by Doxygen 1.8.1.2

Fri May 17 2013 00:36:19

Contents

1	Class Index	1
1.1	Class Hierarchy	1
2	Class Index	3
2.1	Class List	3
3	Class Documentation	5
3.1	frivol::fortune::Algorithm< PolicyT > Class Template Reference	5
3.1.1	Detailed Description	5
3.1.2	Constructor & Destructor Documentation	5
3.1.2.1	Algorithm	6
3.1.3	Member Function Documentation	6
3.1.3.1	getY	6
3.2	frivol::fortune::Arc Struct Reference	6
3.3	frivol::Array< T > Class Template Reference	6
3.3.1	Detailed Description	7
3.3.2	Constructor & Destructor Documentation	7
3.3.2.1	Array	7
3.3.3	Member Function Documentation	7
3.3.3.1	operator[]	7
3.3.3.2	operator[]	7
3.3.3.3	resize	7
3.4	frivol::DummyPriorityQueue< PriorityT > Class Template Reference	8
3.5	frivol::DummySearchTree< ElementT > Class Template Reference	8
3.6	frivol::GeometryTraits< CoordT > Struct Template Reference	8
3.6.1	Detailed Description	9
3.7	frivol::GeometryTraits< double > Struct Template Reference	9
3.8	frivol::GeometryTraits< float > Struct Template Reference	9
3.9	frivol::GeometryTraitsFloat< CoordT > Struct Template Reference	9
3.9.1	Detailed Description	10
3.10	frivol::GeometryTraitsImplementedConcept< CoordT > Class Template Reference	10
3.10.1	Detailed Description	10

3.11	frivol::Point< CoordT > Struct Template Reference	10
3.11.1	Detailed Description	11
3.11.2	Constructor & Destructor Documentation	11
3.11.2.1	Point	11
3.12	frivol::Policy< CoordT, EventQueueT, BeachLineT > Struct Template Reference	11
3.12.1	Detailed Description	11
3.13	frivol::PriorityQueueConcept< X, PriorityT > Class Template Reference	12
3.13.1	Detailed Description	12
3.14	frivol::SearchTreeConcept< X, ElementT > Class Template Reference	12
3.14.1	Detailed Description	13
3.15	frivol::Stack< T > Class Template Reference	13
3.15.1	Detailed Description	13
3.15.2	Member Function Documentation	14
3.15.2.1	push	14
3.15.2.2	top	14

Chapter 1

Class Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

frivol::fortune::Algorithm< PolicyT >	5
frivol::fortune::Arc	6
frivol::Array< T >	6
frivol::DummyPriorityQueue< PriorityT >	8
frivol::DummySearchTree< ElementT >	8
frivol::GeometryTraits< CoordT >	8
frivol::GeometryTraitsFloat< CoordT >	9
frivol::GeometryTraitsFloat< double >	9
frivol::GeometryTraits< double >	9
frivol::GeometryTraitsFloat< float >	9
frivol::GeometryTraits< float >	9
frivol::GeometryTraitsImplementedConcept< CoordT >	10
frivol::Point< CoordT >	10
frivol::Policy< CoordT, EventQueueT, BeachLineT >	11
frivol::PriorityQueueConcept< X, PriorityT >	12
frivol::SearchTreeConcept< X, ElementT >	12
frivol::Stack< T >	13

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

frivol::fortune::Algorithm< PolicyT >	5
frivol::fortune::Arc	
Information of an arc in the beach line	6
frivol::Array< T >	6
frivol::DummyPriorityQueue< PriorityT >	
Simple implementation of PriorityQueueConcept	8
frivol::DummySearchTree< ElementT >	
Simple implementation of SearchTreeConcept (a wrapper around <code>std::list</code>)	8
frivol::GeometryTraits< CoordT >	8
frivol::GeometryTraits< double >	9
frivol::GeometryTraits< float >	9
frivol::GeometryTraitsFloat< CoordT >	9
frivol::GeometryTraitsImplementedConcept< CoordT >	10
frivol::Point< CoordT >	10
frivol::Policy< CoordT, EventQueueT, BeachLineT >	11
frivol::PriorityQueueConcept< X, PriorityT >	12
frivol::SearchTreeConcept< X, ElementT >	12
frivol::Stack< T >	13

Chapter 3

Class Documentation

3.1 `frivol::fortune::Algorithm< PolicyT >` Class Template Reference

```
#include <fortune.hpp>
```

Public Types

- typedef `PolicyT::Coord` **CoordT**
- typedef `Point< CoordT >` **PointT**

Public Member Functions

- `Algorithm` (const `Array< PointT >` &sites)
- void `step` ()
Run the algorithm one event handling forward.
- `CoordT` `getY` () const
- bool `isFinished` ()
Returns true if the algorithm has finished.
- void `finish` ()
Steps the algorithm until the end.
- int `getVoronoiVertexCount` () const
Return the number of Voronoi vertices met in the algorithm.

3.1.1 Detailed Description

```
template<typename PolicyT = DefaultPolicy>class frivol::fortune::Algorithm< PolicyT >
```

State of Fortune's algorithm.

Template Parameters

<code>PolicyT</code>	The algorithm policy to use, instance of <code>Policy</code> template.
----------------------	--

3.1.2 Constructor & Destructor Documentation

3.1.2.1 `template<typename PolicyT > frivol::fortune::Algorithm< PolicyT >::Algorithm (const Array< PointT > & sites)`

Constructs algorithm state.

Parameters

<i>points</i>	Reference to the input set of sites. The object must exist throughout the existence of the Algorithm .
---------------	--

3.1.3 Member Function Documentation

3.1.3.1 `template<typename PolicyT > PolicyT::Coord frivol::fortune::Algorithm< PolicyT >::getY () const`

Get the y coordinate of last [step\(\)](#). Undefined return value if [step\(\)](#) has not been called yet.

The documentation for this class was generated from the following files:

- /home/topi/unison/Asiakirjat/frivol/frivol/fortune.hpp
- /home/topi/unison/Asiakirjat/frivol/frivol/fortune_impl.hpp

3.2 frivol::fortune::Arc Struct Reference

Information of an arc in the beach line.

```
#include <fortune.hpp>
```

Public Attributes

- Idx [site](#)
The site from which the arc originates.
- Idx [arc_id](#)
The ID of the arc.

The documentation for this struct was generated from the following file:

- /home/topi/unison/Asiakirjat/frivol/frivol/fortune.hpp

3.3 frivol::Array< T > Class Template Reference

```
#include <array.hpp>
```

Public Member Functions

- [Array](#) (Idx size)
- Idx [getSize](#) () const
Returns the size of the array.
- void [resize](#) (Idx size)
- const T & [operator\[\]](#) (Idx index) const
- T & [operator\[\]](#) (Idx index)

3.3.1 Detailed Description

template<typename T>class frivol::Array< T >

Simple fixed-size array.

Template Parameters

<i>T</i>	The type of stored elements. Should be default constructible.
----------	---

3.3.2 Constructor & Destructor Documentation

3.3.2.1 template<typename T > frivol::Array< T >::Array (*Idx size*)

Creates an array with all elements default-constructed.

Parameters

<i>size</i>	The size of the array.
-------------	------------------------

3.3.3 Member Function Documentation

3.3.3.1 template<typename T > const T & frivol::Array< T >::operator[] (*Idx index*) const

Returns reference to an element in the array.

Parameters

<i>index</i>	The zero-based index of the element.
--------------	--------------------------------------

Exceptions

<i>std::out_of_range</i>	if FRIVOL_ARRAY_BOUNDS_CHECKING is defined and 'index' overflows.
--------------------------	---

3.3.3.2 template<typename T > T & frivol::Array< T >::operator[] (*Idx index*)

Returns reference to an element in the array.

Parameters

<i>index</i>	The zero-based index of the element.
--------------	--------------------------------------

Exceptions

<i>std::out_of_range</i>	if FRIVOL_ARRAY_BOUNDS_CHECKING is defined and 'index' overflows.
--------------------------	---

3.3.3.3 template<typename T > void frivol::Array< T >::resize (*Idx size*)

Resizes the array to size. If size decreases the extra elements are removed. If size increases, the new elements are default-constructed. The operation may assign the current elements to a new place, and therefore pointers to the array may be invalidated.

Parameters

<i>size</i>	The new size.
-------------	---------------

The documentation for this class was generated from the following files:

- /home/topi/unison/Asiakirjat/frivol/frivol/array.hpp
- /home/topi/unison/Asiakirjat/frivol/frivol/array_impl.hpp

3.4 frivol::DummyPriorityQueue< PriorityT > Class Template Reference

Simple implementation of [PriorityQueueConcept](#).

```
#include <priority_queue_concept.hpp>
```

Public Member Functions

- **DummyPriorityQueue** (Idx size)
- std::pair< Idx, PriorityT > **pop** ()
- bool **empty** () const
- void **setPriority** (Idx key, PriorityT priority)
- void **setPriorityNIL** (Idx key)

The documentation for this class was generated from the following files:

- /home/topi/unison/Asiakirjat/frivol/frivol/priority_queue_concept.hpp
- /home/topi/unison/Asiakirjat/frivol/frivol/priority_queue_concept_impl.hpp

3.5 frivol::DummySearchTree< ElementT > Class Template Reference

Simple implementation of [SearchTreeConcept](#) (a wrapper around std::list).

```
#include <search_tree_concept.hpp>
```

Public Types

- typedef std::list< ElementT >::iterator **Iterator**

Public Member Functions

- template<typename FuncT > Iterator **search** (FuncT func)

The documentation for this class was generated from the following file:

- /home/topi/unison/Asiakirjat/frivol/frivol/search_tree_concept.hpp

3.6 frivol::GeometryTraits< CoordT > Struct Template Reference

```
#include <geometry_traits.hpp>
```

3.6.1 Detailed Description

```
template<typename CoordT>struct frivol::GeometryTraits< CoordT >
```

Traits class that gives needed geometry operations for the algorithm. Implemented traits are required by [Policy](#).

Template Parameters

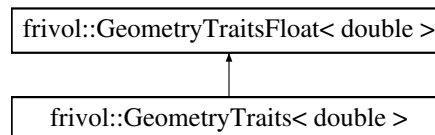
<i>CoordT</i>	The coordinate type to use.
---------------	-----------------------------

The documentation for this struct was generated from the following file:

- /home/topi/unison/Asiakirjat/frivol/frivol/geometry_traits.hpp

3.7 frivol::GeometryTraits< double > Struct Template Reference

Inheritance diagram for frivol::GeometryTraits< double >:



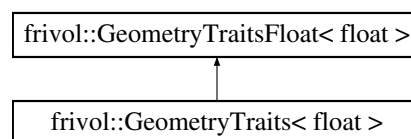
Additional Inherited Members

The documentation for this struct was generated from the following file:

- /home/topi/unison/Asiakirjat/frivol/frivol/geometry_traits.hpp

3.8 frivol::GeometryTraits< float > Struct Template Reference

Inheritance diagram for frivol::GeometryTraits< float >:



Additional Inherited Members

The documentation for this struct was generated from the following file:

- /home/topi/unison/Asiakirjat/frivol/frivol/geometry_traits.hpp

3.9 frivol::GeometryTraitsFloat< CoordT > Struct Template Reference

```
#include <geometry_traits.hpp>
```

Static Public Member Functions

- static CoordT **getBreakpointX** (const Point< CoordT > &a, const Point< CoordT > &b, CoordT topy)
- static CoordT **getCircumcircleTopY** (const Point< CoordT > &a, const Point< CoordT > &b, const Point< CoordT > &c)
- static bool **isCCW** (const Point< CoordT > &a, const Point< CoordT > &b, const Point< CoordT > &c)

3.9.1 Detailed Description

template<typename CoordT>struct frivol::GeometryTraitsFloat< CoordT >

Implementation of [GeometryTraits](#) for floating point coordinate types (float and double).

The documentation for this struct was generated from the following files:

- /home/topi/unison/Asiakirjat/frivol/frivol/geometry_traits.hpp
- /home/topi/unison/Asiakirjat/frivol/frivol/geometry_traits_impl.hpp

3.10 frivol::GeometryTraitsImplementedConcept< CoordT > Class Template Reference

```
#include <geometry_traits.hpp>
```

3.10.1 Detailed Description

template<typename CoordT>class frivol::GeometryTraitsImplementedConcept< CoordT >

Concept for checking that all required [GeometryTraits](#) are implemented for given coordinate type. Required operations are:

- CoordT getBreakpointX(Point<CoordT> a, Point<CoordT> b, CoordT topy) returns the X coordinate of intersection of the two parabolas defined by $(x-a.x)^2 + (y-a.y)^2 = (y-topy)^2 = (x-b.x)^2 + (y-b.y)^2$. The function may assume that $a.x \leq b.x$, $a.y \leq topy$ and $b.y \leq topy$. The function should choose the solution where the parabola around a goes under the parabola around b.
- CoordT getCircumcircleTopY(Point<CoordT> a, Point<CoordT> b, Point<CoordT> c) returns the Y coordinate of the top point (i.e. highest Y coordinate) of the circumscribed circle around triangle 'abc'. In case of (almost) collinear points, the result should be very big or very small compared to site coordinates.
- bool isCCW(Point<CoordT> a, Point<CoordT> b, Point<CoordT> c) returns true if triangle 'abc' is oriented counterclockwise.

Template Parameters

<i>CoordT</i>	The coordinate type.
---------------	----------------------

The documentation for this class was generated from the following file:

- /home/topi/unison/Asiakirjat/frivol/frivol/geometry_traits.hpp

3.11 frivol::Point< CoordT > Struct Template Reference

```
#include <point.hpp>
```

Public Member Functions

- [Point](#) (CoordT [x](#), CoordT [y](#))
- [Point](#) ()

Constructs point with undefined values as coordinates.

Public Attributes

- CoordT [x](#)
The X coordinate of the point.
- CoordT [y](#)
The Y coordinate of the point.

3.11.1 Detailed Description

```
template<typename CoordT = double>struct frivol::Point< CoordT >
```

Two-dimensional point.

Template Parameters

<i>CoordT</i>	The coordinate type to use. Should be default constructible. Defaults to double, which is the coordinate type of DefaultPolicy.
---------------	---

3.11.2 Constructor & Destructor Documentation

```
3.11.2.1  template<typename CoordT = double> frivol::Point< CoordT >::Point ( CoordT x, CoordT y )  [inline]
```

Constructs point with given coordinates.

Parameters

x	The X coordinate.
y	The Y coordinate.

The documentation for this struct was generated from the following file:

- /home/topi/unison/Asiakirjat/frivol/frivol/point.hpp

3.12 frivol::Policy< CoordT, EventQueueT, BeachLineT > Struct Template Reference

```
#include <policy.hpp>
```

Public Types

- typedef CoordT **Coord**

3.12.1 Detailed Description

```
template<typename CoordT, template< typename PriorityT > class EventQueueT, template< typename ElementT > class BeachLineT> struct frivol::Policy< CoordT, EventQueueT, BeachLineT >
```

[Policy](#) class for the Fortune's algorithm, specifying data types and data structures to use.

Template Parameters

<i>CoordT</i>	The coordinate type to use. Should be ordered and default constructible to undefined value. Should have specialization of GeometryTraits .
<i>EventQueueT</i>	The priority queue type for events. Must conform to PriorityQueueConcept .
<i>BeachLineT</i>	The search tree to use for the "beach line" of arcs. Must conform to SearchTreeConcept .

The documentation for this struct was generated from the following file:

- /home/topi/unison/Asiakirjat/frivol/frivol/policy.hpp

3.13 frivol::PriorityQueueConcept< X, PriorityT > Class Template Reference

```
#include <priority_queue_concept.hpp>
```

3.13.1 Detailed Description

```
template<typename X, typename PriorityT> class frivol::PriorityQueueConcept< X, PriorityT >
```

Concept checking class for priority queues *X* with priority values of type *PriorityT* (or *NIL*). Priority queues are initialized with given size, and contain priority values for keys 0, 1, ..., size-1. Initially, all priority values are *NIL*. *X* must support the following operations:

- `<construct>(Idx size)` creates priority queue for keys 0, 1, ..., size-1.
- `bool empty()` const returns true if all keys have *NIL* priority.
- `std::pair<Idx, PriorityT> pop()` returns pair of a key with lowest non-*NIL* priority and its priority and sets the priority to *NIL*.
- `void setPriority(Idx key, PriorityT priority)` sets the priority value of 'key' to non-*NIL* value 'priority'.
- `void setPriorityNIL(Idx key)` sets the priority value of key 'key' to *NIL*.

X may assume that *PriorityT* is ordered with `<`-operator. *X* may have undefined behavior if supplied keys are out of range or if `pop()` is called when `empty()` returns true.

The documentation for this class was generated from the following file:

- /home/topi/unison/Asiakirjat/frivol/frivol/priority_queue_concept.hpp

3.14 frivol::SearchTreeConcept< X, ElementT > Class Template Reference

```
#include <search_tree_concept.hpp>
```

Public Types

- typedef *X::Iterator* **IteratorT**

3.14.1 Detailed Description

```
template<typename X, typename ElementT>class frivol::SearchTreeConcept< X, ElementT >
```

Concept checking class for search trees X for elements of type ElementT. Search trees are sequence containers, the elements of which are iterated using iterator objects of type X::iterator. The iterator must be a standard bidirectional iterator. X must support the following operations:

- `<construct>()` creates empty search tree.
- `bool empty()` const returns true if the search tree is empty.
- Iterator `begin()` returns the iterator of the first element (or past-the-end if empty).
- Iterator `end()` returns the iterator past the last element.
- `template<typename FuncT> Iterator search(FuncT func)` searches the sequence using the supplied int(-iterator)-function that for given iterator iter returns negative if the searched element is before iter, positive if it is after iter, and 0 if iter is the right element. If an element such that func returns 0 is found, it is returned, otherwise `end()` is returned.
- `void erase(iterator iter)` removes element at iter. Other iterators should not be invalidated.
- Iterators `insert(iterator iter, const ElementT& elem)` inserts elem before iter and returns the iterator of the new element. Does not invalidate any iterators.

X may assume that ElementT is copy constructible.

The documentation for this class was generated from the following file:

- `/home/topi/unison/Asiakirjat/frivol/frivol/search_tree_concept.hpp`

3.15 frivol::Stack< T > Class Template Reference

```
#include <stack.hpp>
```

Public Member Functions

- [Stack](#) ()
Constructs empty stack.
- `bool empty ()` const
Returns true if the stack is empty.
- `T & top ()`
- `void pop ()`
Removes the top element of the stack. Call only if [empty\(\)](#) is false.
- `void push (const T &element)`

3.15.1 Detailed Description

```
template<typename T>class frivol::Stack< T >
```

[Stack](#) of elements.

Template Parameters

<i>T</i>	The type of stored elements. Should be default constructible.
----------	---

3.15.2 Member Function Documentation

3.15.2.1 `template<typename T> void frivol::Stack< T >::push (const T & element)`

Pushes element to the top of the stack.

Parameters

<i>element</i>	The element to push.
----------------	----------------------

3.15.2.2 `template<typename T> T & frivol::Stack< T >::top ()`

Returns reference to the top element of the stack. Call only if `empty()` is false.

The documentation for this class was generated from the following files:

- `/home/topi/unison/Asiakirjat/frivol/frivol/stack.hpp`
- `/home/topi/unison/Asiakirjat/frivol/frivol/stack_impl.hpp`